Decorrelating WSN Traffic Patterns with Maximally Uninformative Constrained Routing

Juan E. Tapiador[1], Mudhakar Srivatsa[2],
John A. Clark[1], and John A. McDermid[1]

[1] Department of Computer Science, University of York, York, UK
{jet, jac, jam}@cs.york.ac.uk
[2] IBM Thomas J. Watson Research Center, NY, USA
msrivats@us.ibm.com

Abstract. We study optimal strategies to decorrelating traffic in tactical wireless sensor networks where the goal is hiding sensible information (e.g., communication patterns, nodes location) about ongoing operations implicitly contained in network flows. Contrarily to existing approaches based on heuristic arguments, in this work we pose the problem in a more formal way. In particular, we explore the problem of how to derive routing policies which minimize the path predictability whilst observing certain QoS restrictions. We show how deriving optimal routing strategies can be couched as a nonlinear optimization problem with linear constraints. A convenient reformulation allows us to attack it very efficiently with a numerical least square error solver. Overall, the resulting scheme is an adaptive multipath routing protocol which provides the optimal balance between uninformativeness of routing patterns and end-to-end communication costs.

Keywords: Wireless sensor networks; traffic analysis; security-cost tradeoffs.

1 Introduction

Traffic analysis has traditionally been a major threat to wireless tactical military communications. An adversary with the ability to obtain network measures such as packet counts at various links, correlations in sending and receiving times, etc. may deduce sensitive information about existing communication patterns. Besides the obvious, yet valuable, information about who is communicating with whom and when, such patterns could also be exploited to gain further intelligence. Examples include the physical position and role of some network nodes (e.g., those providing relevant services) and information related to the ongoing mission (e.g. the occurrence of events of interest).

In this work we address some problems related to traffic analysis in wireless sensor networks (WSNs), particularly those used in coalition tactical missions [2, 14]. It is important to stress that these networks present some peculiarities that make them rather different from the traditional hierarchical (i.e. tree-like) WSNs

often deployed in applications such as environmental monitoring. To begin with, the concept of sensor is quite diverse and comprises not only low-capability devices taking simple measures, but also high-end and resource-rich systems such as complex distributed databases and platforms providing audio and video feeds. Some of these nodes may have a relatively fixed position in the terrain, whilst other move around constantly. Much diversity is also found at the application layer. Nodes may engage in a sort of subscriber-publisher model (the likes of Twitter, but richer in content) and social networking applications to form communities of interest and share information. In certain cases, the information delivered by such sensors may have some Quality-of-Service (QoS) constraints. For instance ITU (International Telecommunication Union) recommends up to 250ms one-way latency for voice communication, and recent case studies [13] indicate that latencies over 400ms significantly deteriorate the quality of voice conversations. Further complications arise when such networks are formed by a coalition of forces belonging to different authorities. This introduces not only concerns at the access- and sharing-control level, but also complications in the kind of security services that can be composed (e.g., because of the use of mutually incompatible cryptographic protocols, or simply due to mistrust).

Traffic analysis problems almost equivalent to those found in these environments have received much attention in the field of privacy enhancing technologies, particularly in the form of solutions tailored for mobile ad hoc networks (MANETs). Significant efforts have been devoted to making data transmissions unlikable and hiding network paths through the use of anonymous routing protocols, such as ANODR [6], ASR [20], MASK [19], and ARM [12], to name a few. Unfortunately, these solutions are not appropriate for the WSNs described above for a number of reasons. Many sensors do not have the computational resources required to participate in such protocols. More importantly, different parts of the network will belong to different authorities, some of which may be reticent (or simply unable) to cooperate, either for technical or operational reasons.

Privacy research in traditional WSNs has addressed traffic analysis problems using different, and in some respects more lightweight, techniques (see [8] for a recent survey). Classical problems, such as hiding the location of data sources and sinks, have been attacked from the perspective of providing some degree of unobservability to an adversary. This is achieved by decorrelating the traffic patterns arising from network operations, and a variety of schemes have been explored. We next review the most relevant works in this area and later motivate our work and our main contributions.

1.1 Related work

Various works have addressed the problem of protecting contextual information of WSNs operations by obfuscating traffic patterns. Such patterns ultimately emerge due to: 1) the activity generated by sources and intermediate nodes; and 2) the specific routing paths followed by network flows. The former can be conveniently disguised by a combination of techniques such as padding messages, inserting dummy packets, and a careful selection of transmission rates (e.g.,

uniform, random, or else following a predetermined distribution) [8]. On the other hand, the paths followed by packets towards the destination are ultimately responsible for the statistical traffic patterns observable at a global level.

One common idea in these schemes is randomizing packet routing so as to disguise the actual paths. To this end, the work in [5] suggested a combination of probabilistic flooding and random walk routing. The main idea is to use broadcast to have as many nodes as possible participating in the routing process. This incurs intolerable costs, so it is proposed that each node takes part in the process with some predetermined probability. In random routing approaches, nodes randomly select a neighbor to forward each received packet. This idea is present in some proposals (e.g. [5, 10]) and has proven to be ineffective, as random walks tend to stabilize around the source. In [17] it is proposed to use a hybrid scheme where packets initially follow a random walk until reaching certain nodes; from that point on, packets are routed deterministically.

More sophisticated random walk schemes have been proposed elsewhere. For example, the work in [1] explores a fractal propagation approach where packets fork into multiple paths at some points in the route. All but one of such new paths are fake, in the sense that they carry dummy packets whose only purpose is to introduce confusion. Using both dummy traffic and fake nodes has been further explored in other works (e.g. [5, 9, 18]), in which some strategies to positioning fake sources and generating and filtering out fake traffic are suggested.

1.2 Our contributions

The routing schemes discussed above present several drawbacks, notably:

1. Most of them are tailored to hierarchical WSN architectures. In our scenario, arbitrary topologies may emerge.
2. The routing mechanisms are, in most cases, hugely inefficient in terms of the amount of additional masking traffic introduced. This may impact negatively nodes depending on batteries.
3. They are completely unaware of QoS constraints of (at least some) traffic flows. In some environments, these requirements might be as important as security concerns.
4. In some schemes end-to-end delays can become arbitrarily large, to the extent that there is no guarantee of data delivery to the destination.

In this paper we attempt to overcome these problems by adopting an entirely different approach. At the core of our proposal is a multipath routing protocol similar to those recently developed for MANETs. Each node (sensor) participating in the protocol maintains a local routing table with information about its neighborhood and reachable nodes. The crucial difference with similar schemes lies in the routing policy. In our case, the router obeys a local forwarding policy which maximizes its unpredictability, whilst observing some per-flow QoS constraints. Such a policy is represented by a set of probability distributions, and we show how deriving an optimal one can be reduced to a non-linear optimization problem with linear constraints. We then leverage existing numerical

algorithms to solve the problem. In doing so, our approach quantifies tradeoffs between unpredictability and QoS sensitivity of an application. Furthermore, it can be argued that the policies thus obtained are *provably optimal*, in the sense that they provide the maximum possible uncertainty given current constraints. The scheme is designed to handle network dynamics, both at the topological level (e.g., due to node mobility) and at the application level (changes in user access patterns).

The key contribution of this work lies in deriving optimal routing policies. Due to space reasons we leave open some relevant questions, notably: 1) a detailed description of the multipath routing protocol; 2) a full coverage of how transmission times should be decorrelated; 3) a deep treatment of emergent traffic patterns at a global level, both analytically and through simulations; and 4) a formal security analysis. These will be addressed in an extended version of the paper.

2 Routing model

The core of our proposal is a probabilistic multipath routing scheme in which packets within the same flow may follow different paths to the destination. Each router R maintains a routing table with the structure illustrated in Fig. 1. We will denote by M and N the number of neighbors (i.e., potential next hops) at a given time instant and the number of reachable destinations, respectively. Nodes $n_1, \ldots, n_M$ are the neighbors of R, c_i is the cost (e.g., number of hops, delay, etc.) of reaching d through n_i, and p_i is the probability of forwarding to n_i a packet destined to d. Parameter α_d measures the probability of R receiving a packet destined to d (i.e., the frequency of incoming packets destined to d); the utility for this shall be clear later. For convenience, we will reorganize the routing's main parameters into three elements:

1. $\mathcal{P} = [p_{ij}]$ is an $N \times M$ matrix containing the router's forwarding policy. The value p_{ij} is the probability of sending to n_j a packet destined to d_i. Obviously, $0 \leq p_{ij} \leq 1$ and $\sum_{j=1}^{k} p_{ij} = 1$.
2. $\mathcal{C} = [c_{ij}]$ is an $N \times M$ matrix representing the costs associated with each policy decision. Thus, $c_{ij} \geq 0$ measures the cost of using n_j as next hop for a packet destined to d_i. Unreachable nodes will be associated with an infinite cost, which in our model can be implemented by a sufficiently high cost value. These values are provided by the route discovery and maintenance mechanisms within the routing protocol.
3. $\mathcal{D} = [\alpha_i]$ is the probability distribution of packets destined to d_i arriving at the router. The router can easily (re-)compute them periodically by using a sliding window over the amount of traffic received.

Despite its simplicity, this model is quite flexible and allow us to represent a broad range of different routing policies. For example, a minimum cost (e.g., shortest path) policy can be implemented as

$$\mathcal{P}_{MC} = [p_{ij}] = \begin{cases} 1 & \text{if } c_{ij} \leq c_{ik} \text{ for all } k \neq j \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

Dest.	Flow prob.	Forwarding policy
d_1	α_1	$(n_1, c_{11}, p_{11}) \cdots (n_M, c_{1M}, p_{1M})$
d_2	α_2	$(n_1, c_{21}, p_{21}) \cdots (n_M, c_{2M}, p_{2M})$
$\vdots$	$\vdots$	$\vdots \quad \ddots \quad \vdots$
d_N	α_N	$(n_1, c_{N1}, p_{N1}) \cdots (n_M, c_{NM}, p_{NM})$

Fig. 1. Structure of the routing table.

Similarly, a random walk strategy is given by

$$\mathcal{P}_{RR} = [p_{ij}] = \frac{1}{M} \qquad \text{for all } i, j \tag{2}$$

All the information about a router's observable behavior in terms of link usage is implicitly encoded in $\mathcal{P}$. Subsequently in Section 3 we will address the problem of how to compute the optimal $\mathcal{P}$ while preserving some externally given QoS constraints. Next we provide a brief and informal description on the underlying routing protocol.

2.1 A note on the routing protocol

Multipath routing protocols split traffic over multiple routes in order to maximize network survivability and exploit parallelism in data delivery. These schemes have proliferated recently and a variety of different proposals is available. Our scheme is relatively independent of the underlying multipath routing protocol employed, but some characteristics are highly desirable and will contribute to maximize the amount of traffic decorrelation provided. Here we will focus on multipath routing protocols for MANETs (see [15, 16] for excellent surveys and the references therein contained). The three main characteristics of such protocols are: 1) how to discover multiple paths; 2) how to select which path(s) will be used; and 3) how to distribute the load over the selected path(s). Points 2 and 3 are covered by our routing policies, so we will mainly use route discovery and maintenance mechanisms to obtain and update the routing table. Ideally for our purposes, such mechanisms should provide each router with as much information as possible about available paths to all possible destinations. This will inevitably conflict with other goals as further overhead is introduced. According to [15], the selection of a multipath routing protocol can be done in terms of our expectations in overhead, reliability, delay, energy consumption and mobility support.

Alternatively to using a general purpose multipath routing protocol, some scenarios (e.g., low mobility) might admit a much simpler discovery scheme based on controlled flooding. This will essentially provide each node with a global view of the network topology without all the complexities present in multipath routing protocols, which are unnecessary here.

3 Maximally uninformative constrained policies

The crux of our scheme is the derivation of next-hop forwarding policies such that the aggregated link usage across all flows and all possible next hops provides an observer with as little information as possible about the current path(s) being used. Such policies, however, must observe some externally given constraints in order to limit end-to-end communication costs. Keeping in mind the multipath routing model described in the previous section, our goal is to find the set of distributions $\mathcal{P}$ yielding the maximum possible uncertainty (measured using Shannon entropy) allowed by our constraints. By definition, such a $\mathcal{P}$ will generate the most uninformative link usage that we can afford. In this section we show how this problem can be formally posed as a constrained optimization one and also provide some experimental results obtained with a prototype implementation.

Our constraints will take the form of upper bounds on the expected cost of reaching each destination node. According to the definitions introduced above, the expected cost of reaching d_i is given by $\sum_{j=1}^{M} c_{ij}p_{ij}$. Thus, we will seek a $\mathcal{P}$ such that, for each d_i, this quantity is bounded by a maximum tolerable limit $Q_{d_i}^{max}$. The key advantage of this model is the linearity of the constraints; this will allow us to employ efficient solving techniques. On the other hand, it approximates relatively well the behavior of some QoS metrics (e.g., end-to-end delays as sums of individual delays plus local processing times). We finally make an important remark: by restricting valid policies to those observing a set of constraints, we ensure (in a probabilistic sense) that flows will be delivered to the destination in an upper-bounded amount of time. This guarantees the correctness of the protocol, in the sense that it rules out the occurrence of infinite loops.

3.1 Computing optimal forwarding strategies

Given a routing policy $\mathcal{P} = [p_{ij}]$, the total traffic forwarded through n_i (i.e., the expected usage of link $R \rightarrow n_i$) is given by

$$u_i = \sum_{j=1}^{N} \alpha_j p_{ji} \tag{3}$$

and so we seek a $\mathcal{P}$ such that, for a given $\mathcal{D}$, the Shannon entropy of the expected usage

$$H(\mathcal{P}_{\mathcal{D}}) = -\sum_{i=1}^{M} u_i \log u_i = -\sum_{i=1}^{M} \big(\sum_{j=1}^{N} \alpha_j p_{ji}\big) \log \big(\sum_{j=1}^{N} \alpha_j p_{ji}\big) \tag{4}$$

is maximal, subject to the per-flow QoS constraints given by

$$\forall i = 1, \ldots, N \quad \sum_{j=1}^{M} c_{ij}p_{ij} \leq Q_{d_i}^{max} \tag{5}$$

The nonlinear term in expression (4) prevents us from using a linear optimization solver to obtain $\mathcal{P}$. Despite this, we still can reformulate the problem in a slightly different manner to bring it within the reach of efficient solving techniques.

It is well known that the maximum entropy among discrete probability distributions is achieved by the uniform distribution. Consequently, maximizing $H(\mathcal{P}_\mathcal{D})$ as given by (4) is equivalent to minimizing the deviation of each u_i from the optimal value $\frac{1}{M}$. If we arrange the individual deviations (errors) into a vector $\boldsymbol{e}_{\mathcal{P}_\mathcal{D}} = (e_i)^T$ such that

$$e_i = \left(\sum_{j=1}^{N} \alpha_j p_{ji} \right) - \frac{1}{M} \tag{6}$$

then, following a classical least square error (LSE) approach, the objective can be rewritten as

$$\min_{\mathcal{P}} \frac{1}{2} \parallel \boldsymbol{e}_{\mathcal{P}_\mathcal{D}} \parallel_2^2 \tag{7}$$

subject to the linear constraints mentioned above, plus those needed to ensure that $\mathcal{P}$ is a set of valid probability distributions (that is, $0 \leq p_{ij} \leq 1$ and $\sum_{j=1}^{M} p_{ij} = 1$). A full description of the problem in canonical form is provided in Appendix A. Thus formulated, deriving optimal routing policy reduces to solving a constrained linear least-squares problem. Many numerical methods can be used for this, especially those based on trust-region approaches [3]. In general, the solution is not guaranteed to exist, as the problem may be ill-conditioned or simply be infeasible with given constraints. Available solving techniques can detect if this is the case or otherwise provide us with the optimal (or nearly optimal) solution.

A detailed description of the solving algorithm is out of the scope of this work, yet some particularities are worth mentioning. Most numerical solvers used to attack problems of this sort work in two phases. An initial feasible solution (if it exists) is first computed. The core of the procedure is a loop where a sequence of feasible solutions converging to the optimum is produced [4]. If the problem is feasible, the algorithm iterates until: (i) an optimum is found; or (ii) the number of iterations exceeds a predefined limit; or (iii) no further progress can be made (e.g., due to ill-conditioning, changes smaller than predefined tolerance, etc.) In all cases, the solver returns the best solution found so far. If the error is still too large for our taste, the method can be called again using the obtained solution as starting point, in the hope that a few more iterations will improve it. This scheme is particularly well suited for applications where time constraints apply to the policy (re-)computation process, as it will deliver the best available solution given current computation resources.

3.2 Recomputing existing policies

Changes in network topology and/or traffic flow patterns will require the node to recompute the existing policy. In the case of traffic flows, this can be detected by

measuring a substantial change between the actual flow distribution $\mathcal{D}$ and the one used the last time the policy was obtained. A variety of easily computable metrics can be used for this purpose, particularly the well-known Kullback-Leibler divergence [7]. On the other hand, changes in network topology will be dealt with by the underlying route maintenance protocol, which should keep updated the information contained in the routing tables.

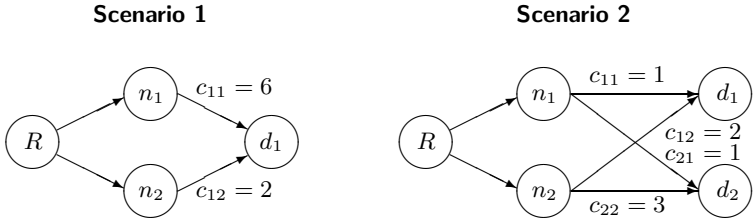
3.3 Implementation and examples

We have implemented a proof-of-concept prototype in MATLAB based on the `lsqlin` solver available in the Optimization Toolbox [11]. This essentially requires us to rewrite the problem in canonical form as specified in Appendix A. Alternatively, a variety of solvers available in existing libraries can be used if a faster implementation is required.

We next provide some examples of optimal routing policies derived for a few simple cases. Consider the first scenario shown in Fig. 2. Here there is just one destination node (d_1) and two possible forwarding nodes (n_1 and n_2), with associated costs 6 and 2, respectively. If the maximum tolerable cost is 2, the only routing policy satisfying the constraints consist of always using n_2 as next node for d_1 (see first row in the table shown in Fig. 2). This policy is completely deterministic and therefore provides zero entropy. If the affordable cost increases to 3, then n_1 can be stochastically used 25% of the time. This preserves the maximum expected cost whilst raising the policy entropy up to 0.811 (the maximum attainable entropy in this case is 1). A further increase in $Q_{d_1}^{max}$ up to 4 would allow a totally uninformative policy where both nodes are equally used. Note too how additional relaxations on the constraints do not translate in a different policy, for this is optimal. The second scenario illustrates a slightly more complex topology where two neighbors (n_1 and n_2) can be used to forward traffic to two destinations (d_1 and d_2) which receive the same fraction of packets ($\alpha_1 = \alpha_2 = 0.5$). As before, very stringent constraints in the expected routing costs would translate in deterministic forwarding strategies with zero entropy (see first row in the table). Successive increments in the tolerable costs give room for more uninformativeness, eventually yielding optimality when possible.

4 Complexity analysis and empirical results

The routing table at each node requires $O(N(2+3M))$ of memory. Even though this is a linear increase with respect to classical routing schemes, we do not anticipate it will constitute a problem for most platforms. Thus for example, in a network with 500 nodes and an average neighborhood of size 25, the routing table will require around 150 KB (assuming 32-bit numbers). The time complexity depends entirely on the method used to derive the policy. In our implementation, the core of the procedure is a quadratic programming solver with linear constraints. Depending on the characteristics of the specific instance (particularly, on the definiteness of the quadratic term matrix), the running time can, in most cases, be polynomially bounded.



Flow prob.	Cost	Constraints	Optimal policy	Entropy
Scenario 1				
$\mathcal{D} = (1)$	$\mathcal{C} = (6\ 2)$	$Q_d^{max} = (2)$	$\mathcal{P} = (0\ 1)$	$H(\mathcal{P}_\mathcal{D}) = 0$
$\mathcal{D} = (1)$	$\mathcal{C} = (6\ 2)$	$Q_d^{max} = (3)$	$\mathcal{P} = (0.25\ 0.75)$	$H(\mathcal{P}_\mathcal{D}) = 0.811$
$\mathcal{D} = (1)$	$\mathcal{C} = (6\ 2)$	$Q_d^{max} = (4)$	$\mathcal{P} = (0.5\ 0.5)$	$H(\mathcal{P}_\mathcal{D}) = 1$
$\mathcal{D} = (1)$	$\mathcal{C} = (6\ 2)$	$Q_d^{max} = (5)$	$\mathcal{P} = (0.5\ 0.5)$	$H(\mathcal{P}_\mathcal{D}) = 1$
Scenario 2				
$\mathcal{D} = \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}$	$\mathcal{C} = \begin{pmatrix} 1 & 2 \\ 1 & 3 \end{pmatrix}$	$Q_d^{max} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$	$\mathcal{P} = \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix}$	$H(\mathcal{P}_\mathcal{D}) = 0$
$\mathcal{D} = \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}$	$\mathcal{C} = \begin{pmatrix} 1 & 2 \\ 1 & 3 \end{pmatrix}$	$Q_d^{max} = \begin{pmatrix} 1.5 \\ 1 \end{pmatrix}$	$\mathcal{P} = \begin{pmatrix} 0.5 & 0.5 \\ 1 & 0 \end{pmatrix}$	$H(\mathcal{P}_\mathcal{D}) = 0.811$
$\mathcal{D} = \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}$	$\mathcal{C} = \begin{pmatrix} 1 & 2 \\ 1 & 3 \end{pmatrix}$	$Q_d^{max} = \begin{pmatrix} 1.5 \\ 2 \end{pmatrix}$	$\mathcal{P} = \begin{pmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{pmatrix}$	$H(\mathcal{P}_\mathcal{D}) = 1$
$\mathcal{D} = \begin{pmatrix} 0.2 \\ 0.8 \end{pmatrix}$	$\mathcal{C} = \begin{pmatrix} 1 & 2 \\ 1 & 3 \end{pmatrix}$	$Q_d^{max} = \begin{pmatrix} 1 \\ 1.5 \end{pmatrix}$	$\mathcal{P} = \begin{pmatrix} 1 & 0 \\ 0.75 & 0.25 \end{pmatrix}$	$H(\mathcal{P}_\mathcal{D}) = 0.722$

Fig. 2. Examples of derived optimal routing strategies for two simple scenarios.

Fig. 3 (left) shows the average computing time required to obtain a policy. The results are averaged over 50 randomly generated problem instances and show a polynomial increase in both N and M. These times are reasonable for the scenarios where we expect our scheme to be deployed. Furthermore, we expect that a dedicated implementation in a more efficient language (as opposed to the general purpose `lsqlin` used here) will increase the efficiency in at least one order of magnitude.

Given the iterative nature of the numerical solver, another primary question is the quality of the solutions provided when the procedure stops after reaching the maximum number of iterations without having stabilized in an optimum. To the best of our knowledge, no analytical insight can be used here, and the loop length has to be adjusted manually. Fig. 3 (right) shows the quality of the solutions found when the number of iterations is fixed to 200. This setting is the same used to obtain the curves shown in 3 (left) and was found appropriate by empirical investigation. Here we measure the relative entropy (i.e. actual entropy divided by the maximum attainable entropy) of the solutions obtained. For the values of N and M studied, 200 iterations suffices to obtain policies

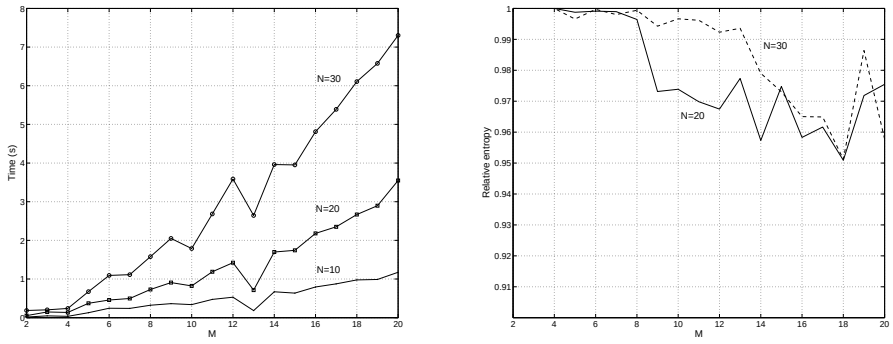


Fig. 3. Average computing time and solution optimality.

whose entropy is at least 95% of the maximum possible. We stress that in most cases the quality of the policy can be increased by additional iterations. Thus, in scenarios wherein a certain entropy threshold should be guaranteed, the solver can be repeatedly called until reaching the sought quality level.

5 Conclusions

We have revisited the problem of designing routing mechanisms aimed at decorrelating traffic patterns in WSNs. Such schemes constitute a fundamental building block to countering traffic analysis attacks, particularly those focused on compromising location privacy.

Prior work on this area has mostly relied on heuristic strategies to hide the actual paths followed by traffic flows. In this paper we have shown how this problem admits a simple and elegant formulation as a maximization problem in a multipath routing setting. By doing so, we can also impose constraints in the expected performance on a per-flow basis. This helps to accommodate some QoS requirements often found in military environments. Obtaining optimal policies thus reduces to solving a constrained LSE problem, for which efficient numerical methods are available. Our simulations show that computing optimal policies can be done efficiently even by nodes with limited computational resources.

A number of relevant details (enumerated in the introduction) have been omitted from this paper due to space constraints. These shall be conveniently addressed in an extended version of this work.

Acknowledgments

This research is sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of

A Problem representation in canonical form

Next we detail how the problem of computing the optimal constrained policy can be rewritten as an LSE problem in canonical form, i.e., as

$$\min_{\boldsymbol{x}} \parallel \boldsymbol{C}\boldsymbol{x} - \boldsymbol{d} \parallel_2^2 \quad \text{s.t.} \quad \begin{cases} \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b} \\ \boldsymbol{A_{eq}}\boldsymbol{x} = \boldsymbol{b_{eq}} \\ \boldsymbol{b_l} \leq \boldsymbol{x} \leq \boldsymbol{b_u} \end{cases} \tag{8}$$

The policy $\mathcal{P} = [p_{ij}]$ must be rearranged into vector form as follows:

$$\boldsymbol{x} = (p_{11}, \ldots, p_{N1}, p_{12}, \ldots, p_{N2}, p_{1M}, \ldots, p_{NM})^T \tag{9}$$

Matrix $\boldsymbol{C}$ contains the flow weights conveniently arranged

$$\boldsymbol{C} = \begin{pmatrix} \alpha_1 \cdots \alpha_N & 0 \cdots & 0 & \cdots & 0 \cdots & 0 \\ 0 \cdots 0 & \alpha_1 \cdots \alpha_N & \cdots & 0 \cdots & 0 \\ \vdots \ddots \vdots & \vdots \ddots \vdots & \ddots & \vdots \ddots \vdots & \vdots \\ 0 \cdots 0 & 0 \cdots 0 & \cdots & \alpha_1 \cdots \alpha_N & \end{pmatrix} \tag{10}$$

and vector $\boldsymbol{d}$ is simply the uniform distribution

$$\boldsymbol{d} = (\frac{1}{M}, \ldots, \frac{1}{M})^T \tag{11}$$

The first set of constraints ensure that each row of $\mathcal{P}$ is a probability distribution. Thus, the two bound vectors encode the fact that $p_{ij} \in [0, 1]$

$$\boldsymbol{b_l} = (0, \ldots, 0)^T \qquad \boldsymbol{b_u} = (1, \ldots, 1)^T \tag{12}$$

while $\boldsymbol{A_{eq}}$ and $\boldsymbol{b_{eq}}$ ensure that each set of probabilities must add up to one

$$\boldsymbol{A_{eq}} = \begin{pmatrix} 1\,0\,0\cdots 0 & 1\,0\,0\cdots 0 & \cdots & 1\,0\,0\cdots 0 \\ 0\,1\,0\cdots 0 & 0\,1\,0\cdots 0 & \cdots & 0\,1\,0\cdots 0 \\ \vdots\,\vdots\,\vdots\ddots & \vdots\,\vdots\,\vdots\ddots & \ddots & \vdots\,\vdots\,\vdots\ddots \\ 0\,0\,0\cdots 1 & 0\,0\,0\cdots 1 & \cdots & 0\,0\,0\cdots 1 \end{pmatrix} \tag{13}$$

$$\boldsymbol{b_{eq}} = (1, \ldots, 1)^T \tag{14}$$

Finally, $\boldsymbol{A}$ and $\boldsymbol{b}$ encode the cost-related linear constraints

$$\boldsymbol{A} = \begin{pmatrix} c_{11} & 0 & 0\cdots & 0 & c_{12} & 0 & 0\cdots & 0 & \cdots & c_{1M} & 0 & 0\cdots & 0 \\ 0 & c_{21} & 0\cdots & 0 & 0 & c_{22} & 0\cdots & 0 & \cdots & 0 & c_{2M} & 0\cdots & 0 \\ \vdots & \vdots & \vdots\ddots & \vdots & \vdots & \vdots & \vdots\ddots & \vdots & \ddots & \vdots & \vdots & \vdots\ddots & \vdots \\ 0 & 0 & 0\cdots & c_{N1} & 0 & 0 & 0\cdots & c_{N2} & \cdots & 0 & 0 & 0\cdots & c_{NM} \end{pmatrix} \tag{15}$$

$$\boldsymbol{b} = (Q_{d_1}^{max}, \ldots, Q_{d_N}^{max})^T \tag{16}$$

References

1. J. Deng, R. Han, and S. Mishra. "Decorrelating wireless sensor network traffic to inhibit traffic analysis attacks." *Pervasive and Mobile Computing*, 2:159–186, 2006.
2. G.F. Elmasry. "A comparative review of commercial vs. tactical wireless networks." *IEEE Communications Magazine* 48(10):54–59, 2010.
3. P.E. Gill, W. Murray, M.H. Wright. *Practical Optimization*, Academic Press, 1981.
4. P.E. Gill, W. Murray, M.A. Saunders, and M.H. Wright. "Procedures for Optimization Problems with a Mixture of Bounds and General Linear Constraints," *ACM Trans. Math. Software*, Vol. 10, pp 282–298, 1984.
5. P. Kamat, Y.Y. Zhang, W. Trappe, and C. Ozturk. "Enhancing source-location privacy in sensor network routing." In *ICDCS 2005*, pp. 599–608.
6. J. Kong and X. Hong. "ANODR: Anonymous on demand routing protocol with untraceable routes for mobile ad-hoc networks." In *ACM MobiHoc*, 2003.
7. S. Kullback and R.A. Leibler. "On Information and Sufficiency." *Annals of Mathematical Statistics*, 22 (1): 79–86, 1951.
8. N. Li, N. Zhang, S.K. Das, and B. Thuraisingham. "Privacy preservation in wireless sensor networks: A state-of-the-art survey." *Ad Hoc Networks* 7:1501–1514, 2009.
9. K. Mehta, D.G. Liu, M. Wright. "Location privacy in sensor networks against a global eavesdropper." In *ICNP 2007*, 314–323.
10. E.C.-H. Ngai. "On providing sink anonymity for sensor networks." In *IWCMC 2009*, 269–273.
11. *Optimization Toolbox User's Guide, Version 5*. The MathWorks, Inc., 2010.
12. S. Seys and B. Preneel. "ARM: Anonymous Routing Protocol for Mobile Ad Hoc Networks." *Intl. J. Wireless and Mobile Computing*, 3(3):145–155, 2009.
13. G. I. Sound. "VoIP: Better than PSTN?." Available online at: `http://www.globalipsound.com/demo/tutorial.php`.
14. N. Suri, G. Benincasa, M. Tortonesi, C. Stefanelli, J. Kovach, R. Winkler, R. Kohler, J. Hanna, L. Pochet, and S. Watson. "Peer-to-peer communications for tactical environments: Observations, requirements, and experiences." *IEEE Communications Magazine* 48(10):60–69, 2010.
15. M. Tarique, K.E. Tepe, S. Adibi, and S. Erfani; "Survey of multipath routing protocols for mobile ad hoc networks." *J. Network and Computer Applications*, 32:1125–1143, 2009.
16. A. Tsirigos and Z. J. Haas. "Multipath Routing in the Presence of Frequent Topological Changes." *IEEE Communications Magazine*, 39(11):132–138, 2001.
17. Y. Xi, L. Schwiebert, W.S. Shi. "Preserving source location privacy in monitoring-based wireless sensor networks." In *IPDPS 2006*.
18. Y. Yang, M. Shao, S. Zhu, B. Urgaonkar, G. Cao. "Towards event source unobservability with minimum network traffic in sensor networks." *WiSec'08*, pp. 77-88.
19. Y. Zhang, W. Liu, and W. Lou. "Anonymous communications in mobile ad hoc networks." In *INFOCOM 2005*.
20. B. Zhu, Z. Wan, M. Kankanhalli, F. Bao, and R. Deng. "Anonymous secure routing in mobile ad-hoc networks." In *LCN 2004*, pp. 102–108, 2004.